CS425 – DATABASE ORGANIZATION

PROJECT DELIVERABLE 2 – Create and load data into the database.

Team Members

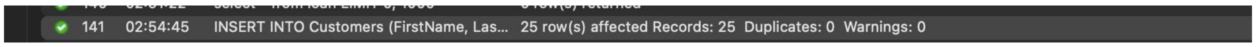
Name	CWID
Vivek Saka	A20554037
Bhavana Pandey	A20554338
Akhil Kumar Marni	A20554334
Akash Thirumuruganantham	A20539883

1. **Inserting data into tables :**

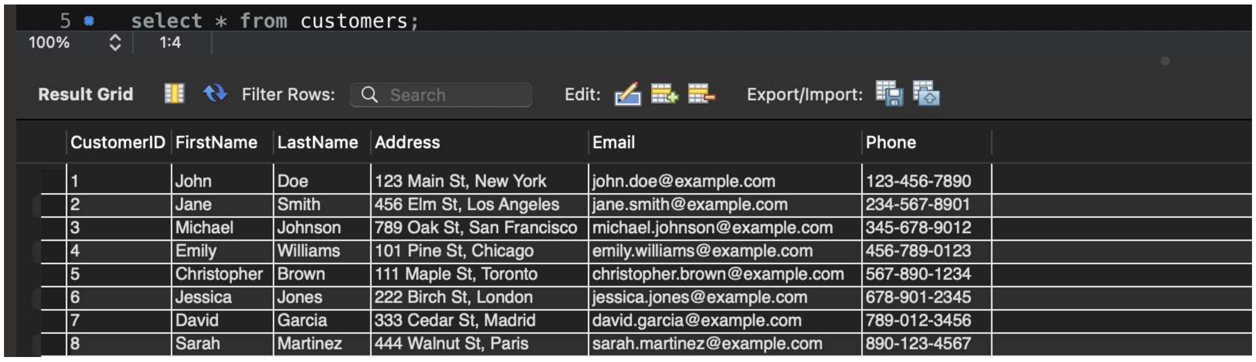
   a. **Customers Table:**
   SQL :Insert statements for Customers table

```
INSERT INTO Customers (FirstName, LastName, Address, Email, Phone) VALUES
('John', 'Doe', '123 Main St, New York', 'john.doe@example.com', '123-456-7890'),
('Jane', 'Smith', '456 Elm St, Los Angeles', 'jane.smith@example.com', '234-567-
8901'),
('Michael', 'Johnson', '789 Oak St, San Francisco', 'michael.johnson@example.com',
'345-678-9012'),
('Emily', 'Williams', '101 Pine St, Chicago', 'emily.williams@example.com', '456-789-
0123'),
('Christopher', 'Brown', '111 Maple St, Toronto', 'christopher.brown@example.com',
'567-890-1234'),
('Jessica', 'Jones', '222 Birch St, London', 'jessica.jones@example.com', '678-901-
2345'),
('David', 'Garcia', '333 Cedar St, Madrid', 'david.garcia@example.com', '789-012-
3456'),
('Sarah', 'Martinez', '444 Walnut St, Paris', 'sarah.martinez@example.com', '890-123-
4567'),
('Andrew', 'Hernandez', '555 Spruce St, Tokyo', 'andrew.hernandez@example.com',
'901-234-5678'),
('Lisa', 'Lopez', '666 Cherry St, Sydney', 'lisa.lopez@example.com', '012-345-6789'),
('Daniel', 'Young', '777 Poplar St, Moscow', 'daniel.young@example.com', '123-456-
7890'),
('Mary', 'King', '888 Sycamore St, Beijing', 'mary.king@example.com', '234-567-
8901'),
('James', 'Lee', '999 Ash St, Melbourne', 'james.lee@example.com', '345-678-9012'),
('Jennifer', 'Scott', '222 Pine Ave, Toronto', 'jennifer.scott@example.com', '456-789-
0123'),
('Matthew', 'Green', '333 Elm Ave, New York', 'matthew.green@example.com', '567-
890-1234'),
('Patricia', 'Adams', '444 Oak Ave, Los Angeles', 'patricia.adams@example.com', '678-
901-2345'),
('Richard', 'Baker', '555 Maple Ave, San Francisco', 'richard.baker@example.com',
'789-012-3456'),
('Elizabeth', 'Gonzalez', '666 Birch Ave, Chicago', 'elizabeth.gonzalez@example.com',
'890-123-4567'),
('William', 'Nelson', '777 Cedar Ave, Toronto', 'william.nelson@example.com', '901-
234-5678'),
('Linda', 'Carter', '888 Walnut Ave, London', 'linda.carter@example.com', '012-345-
6789'),
('Charles', 'Mitchell', '999 Spruce Ave, Madrid', 'charles.mitchell@example.com',
'123-456-7890'),
('Karen', 'Perez', '111 Cherry Ave, Paris', 'karen.perez@example.com', '234-567-
8901'),
('Mark', 'Roberts', '222 Poplar Ave, Tokyo', 'mark.roberts@example.com', '345-678-
9012'),
('Barbara', 'Turner', '333 Sycamore Ave, Sydney', 'barbara.turner@example.com',
'456-789-0123'),
('Joseph', 'Phillips', '444 Ash Ave, Moscow', 'joseph.phillips@example.com', '567-
890-1234');
```

Output :



```
141   02:54:45   INSERT INTO Customers (FirstName, Las...   25 row(s) affected Records: 25  Duplicates: 0  Warnings: 0
```



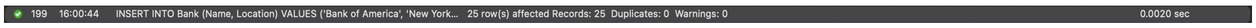
```
select * from customers;
```

CustomerID	FirstName	LastName	Address	Email	Phone
1	John	Doe	123 Main St, New York	john.doe@example.com	123-456-7890
2	Jane	Smith	456 Elm St, Los Angeles	jane.smith@example.com	234-567-8901
3	Michael	Johnson	789 Oak St, San Francisco	michael.johnson@example.com	345-678-9012
4	Emily	Williams	101 Pine St, Chicago	emily.williams@example.com	456-789-0123
5	Christopher	Brown	111 Maple St, Toronto	christopher.brown@example.com	567-890-1234
6	Jessica	Jones	222 Birch St, London	jessica.jones@example.com	678-901-2345
7	David	Garcia	333 Cedar St, Madrid	david.garcia@example.com	789-012-3456
8	Sarah	Martinez	444 Walnut St, Paris	sarah.martinez@example.com	890-123-4567

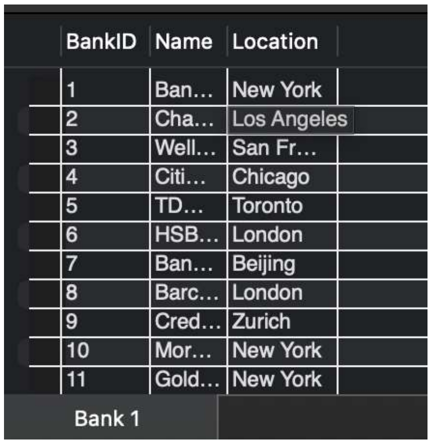
b. **Bank Table :**
SQL :Insert statements for Bank table:

```
INSERT INTO Bank (Name, Location) VALUES
('Bank of America', 'New York'),
('Chase Bank', 'Los Angeles'),
('Wells Fargo', 'San Francisco'),
('Citi Bank', 'Chicago'),
('TD Bank', 'Toronto'),
('HSBC Bank', 'London'),
('Bank of China', 'Beijing'),
('Barclays', 'London'),
('Credit Suisse', 'Zurich'),
('Morgan Stanley', 'New York'),
('Goldman Sachs', 'New York'),
('UBS', 'Zurich'),
('Banco Santander', 'Madrid'),
('BNP Paribas', 'Paris'),
('Royal Bank of Canada', 'Toronto'),
('Mitsubishi UFJ Financial Group', 'Tokyo'),
('Bank of Communications', 'Shanghai'),
('Commonwealth Bank', 'Sydney'),
('Sberbank', 'Moscow'),
('ICBC', 'Beijing'),
('ANZ Bank', 'Melbourne'),
('Westpac Banking Corporation', 'Sydney'),
('NAB', 'Melbourne'),
('Bank of Montreal', 'Toronto'),
('Scotiabank', 'Toronto');
```

Output :



```
199  16:00:44  INSERT INTO Bank (Name, Location) VALUES ('Bank of America', 'New York...  25 row(s) affected Records: 25  Duplicates: 0  Warnings: 0  0.0020 sec
```



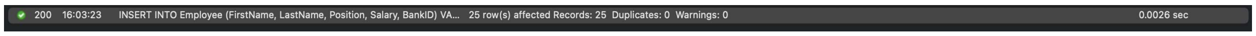
BankID	Name	Location
1	Ban…	New York
2	Cha…	Los Angeles
3	Well…	San Fr…
4	Citi…	Chicago
5	TD…	Toronto
6	HSB…	London
7	Ban…	Beijing
8	Barc…	London
9	Cred…	Zurich
10	Mor…	New York
11	Gold…	New York

c. **Employee Table:**

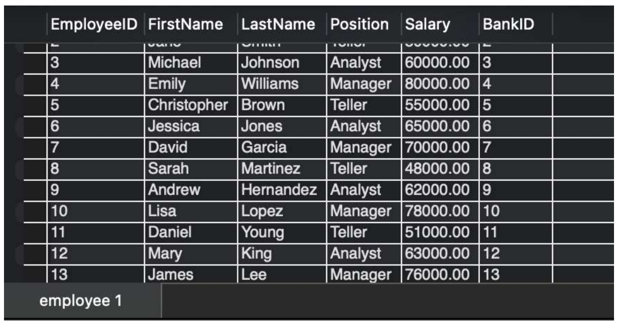
SQL : Insert statements for Employee table

```
INSERT INTO Employee (FirstName, LastName, Position, Salary, BankID) VALUES
('John', 'Doe', 'Manager', 75000.00, 1),
('Jane', 'Smith', 'Teller', 50000.00, 2),
('Michael', 'Johnson', 'Analyst', 60000.00, 3),
('Emily', 'Williams', 'Manager', 80000.00, 4),
('Christopher', 'Brown', 'Teller', 55000.00, 5),
('Jessica', 'Jones', 'Analyst', 65000.00, 6),
('David', 'Garcia', 'Manager', 70000.00, 7),
('Sarah', 'Martinez', 'Teller', 48000.00, 8),
('Andrew', 'Hernandez', 'Analyst', 62000.00, 9),
('Lisa', 'Lopez', 'Manager', 78000.00, 10),
('Daniel', 'Young', 'Teller', 51000.00, 11),
('Mary', 'King', 'Analyst', 63000.00, 12),
('James', 'Lee', 'Manager', 76000.00, 13),
('Jennifer', 'Scott', 'Teller', 49000.00, 14),
('Matthew', 'Green', 'Analyst', 64000.00, 15),
('Patricia', 'Adams', 'Manager', 77000.00, 16),
('Richard', 'Baker', 'Teller', 52000.00, 17),
('Elizabeth', 'Gonzalez', 'Analyst', 66000.00, 18),
('William', 'Nelson', 'Manager', 74000.00, 19),
('Linda', 'Carter', 'Teller', 47000.00, 20),
('Charles', 'Mitchell', 'Analyst', 61000.00, 21),
('Karen', 'Perez', 'Manager', 79000.00, 22),
('Mark', 'Roberts', 'Teller', 53000.00, 23),
('Barbara', 'Turner', 'Analyst', 67000.00, 24),
('Joseph', 'Phillips', 'Manager', 73000.00, 25);
```

Output :



```
200  16:03:23  INSERT INTO Employee (FirstName, LastName, Position, Salary, BankID) VA...  25 row(s) affected Records: 25  Duplicates: 0  Warnings: 0  0.0026 sec
```



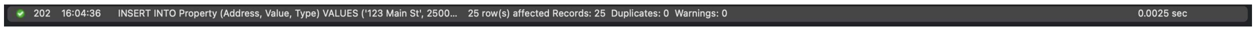
EmployeeID	FirstName	LastName	Position	Salary	BankID
3	Michael	Johnson	Analyst	60000.00	3
4	Emily	Williams	Manager	80000.00	4
5	Christopher	Brown	Teller	55000.00	5
6	Jessica	Jones	Analyst	65000.00	6
7	David	Garcia	Manager	70000.00	7
8	Sarah	Martinez	Teller	48000.00	8
9	Andrew	Hernandez	Analyst	62000.00	9
10	Lisa	Lopez	Manager	78000.00	10
11	Daniel	Young	Teller	51000.00	11
12	Mary	King	Analyst	63000.00	12
13	James	Lee	Manager	76000.00	13

d. **Property Table :**

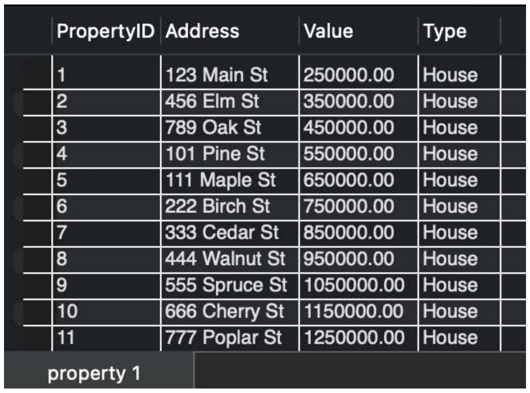
SQL : Insert statements for Property table

```
INSERT INTO Property (Address, Value, Type) VALUES
('123 Main St', 250000.00, 'House'),
('456 Elm St', 350000.00, 'House'),
('789 Oak St', 450000.00, 'House'),
('101 Pine St', 550000.00, 'House'),
('111 Maple St', 650000.00, 'House'),
('222 Birch St', 750000.00, 'House'),
('333 Cedar St', 850000.00, 'House'),
('444 Walnut St', 950000.00, 'House'),
('555 Spruce St', 1050000.00, 'House'),
('666 Cherry St', 1150000.00, 'House'),
('777 Poplar St', 1250000.00, 'House'),
('888 Sycamore St', 1350000.00, 'House'),
('999 Ash St', 1450000.00, 'House'),
('222 Pine Ave', 250000.00, 'Apartment'),
('333 Elm Ave', 350000.00, 'Apartment'),
('444 Oak Ave', 450000.00, 'Apartment'),
('555 Maple Ave', 550000.00, 'Apartment'),
('666 Birch Ave', 650000.00, 'Apartment'),
('777 Cedar Ave', 750000.00, 'Apartment'),
('888 Walnut Ave', 850000.00, 'Apartment'),
('999 Spruce Ave', 950000.00, 'Apartment'),
('111 Cherry Ave', 1050000.00, 'Apartment'),
('222 Poplar Ave', 1150000.00, 'Apartment'),
('333 Sycamore Ave', 1250000.00, 'Apartment'),
('444 Ash Ave', 1350000.00, 'Apartment');
```

Output :



```
202   16:04:36   INSERT INTO Property (Address, Value, Type) VALUES ('123 Main St', 2500...   25 row(s) affected Records: 25  Duplicates: 0  Warnings: 0        0.0025 sec
```



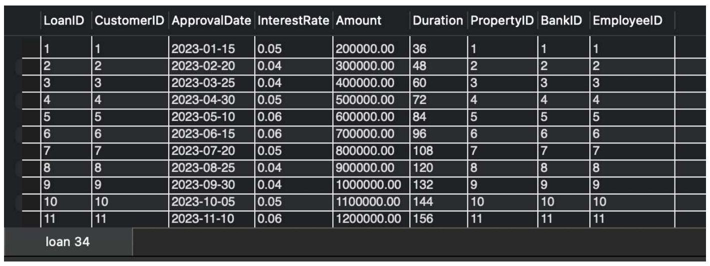
PropertyID	Address	Value	Type
1	123 Main St	250000.00	House
2	456 Elm St	350000.00	House
3	789 Oak St	450000.00	House
4	101 Pine St	550000.00	House
5	111 Maple St	650000.00	House
6	222 Birch St	750000.00	House
7	333 Cedar St	850000.00	House
8	444 Walnut St	950000.00	House
9	555 Spruce St	1050000.00	House
10	666 Cherry St	1150000.00	House
11	777 Poplar St	1250000.00	House

e. **Loan table:**

SQL :Insert statements for Loan table.

```
INSERT INTO Loan (CustomerID, ApprovalDate, InterestRate, Amount, Duration,
PropertyID, BankID, EmployeeID) VALUES
(1, '2023-01-15', 0.045, 200000.00, 36, 1, 1, 1),
(2, '2023-02-20', 0.04, 300000.00, 48, 2, 2, 2),
(3, '2023-03-25', 0.035, 400000.00, 60, 3, 3, 3),
(4, '2023-04-30', 0.05, 500000.00, 72, 4, 4, 4),
(5, '2023-05-10', 0.055, 600000.00, 84, 5, 5, 5),
(6, '2023-06-15', 0.06, 700000.00, 96, 6, 6, 6),
(7, '2023-07-20', 0.045, 800000.00, 108, 7, 7, 7),
(8, '2023-08-25', 0.04, 900000.00, 120, 8, 8, 8),
(9, '2023-09-30', 0.035, 1000000.00, 132, 9, 9, 9),
(10, '2023-10-05', 0.05, 1100000.00, 144, 10, 10, 10),
(11, '2023-11-10', 0.055, 1200000.00, 156, 11, 11, 11),
(12, '2023-12-15', 0.06, 1300000.00, 168, 12, 12, 12),
(13, '2024-01-20', 0.045, 1400000.00, 180, 13, 13, 13),
(14, '2024-02-25', 0.04, 1500000.00, 192, 14, 14, 14),
(15, '2024-03-01', 0.035, 1600000.00, 204, 15, 15, 15),
(16, '2024-04-05', 0.05, 1700000.00, 216, 16, 16, 16),
(17, '2024-05-10', 0.055, 1800000.00, 228, 17, 17, 17),
(18, '2024-06-15', 0.06, 1900000.00, 240, 18, 18, 18),
(19, '2024-07-20', 0.045, 2000000.00, 252, 19, 19, 19),
(20, '2024-08-25', 0.04, 2100000.00, 264, 20, 20, 20),
(21, '2024-09-30', 0.035, 2200000.00, 276, 21, 21, 21),
(22, '2024-10-05', 0.05, 2300000.00, 288, 22, 22, 22),
(23, '2024-11-10', 0.055, 2400000.00, 300, 23, 23, 23),
(24, '2024-12-15', 0.06, 2500000.00, 312, 24, 24, 24),
(25, '2025-01-20', 0.045, 2600000.00, 324, 25, 25, 25);
```

Output :



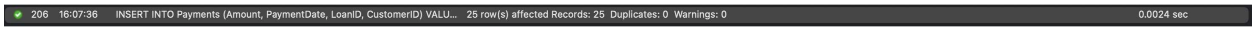
LoanID	CustomerID	ApprovalDate	InterestRate	Amount	Duration	PropertyID	BankID	EmployeeID
1	1	2023-01-15	0.05	200000.00	36	1	1	1
2	2	2023-02-20	0.04	300000.00	48	2	2	2
3	3	2023-03-25	0.04	400000.00	60	3	3	3
4	4	2023-04-30	0.05	500000.00	72	4	4	4
5	5	2023-05-10	0.06	600000.00	84	5	5	5
6	6	2023-06-15	0.06	700000.00	96	6	6	6
7	7	2023-07-20	0.05	800000.00	108	7	7	7
8	8	2023-08-25	0.04	900000.00	120	8	8	8
9	9	2023-09-30	0.04	1000000.00	132	9	9	9
10	10	2023-10-05	0.05	1100000.00	144	10	10	10
11	11	2023-11-10	0.06	1200000.00	156	11	11	11

f. Payments table :

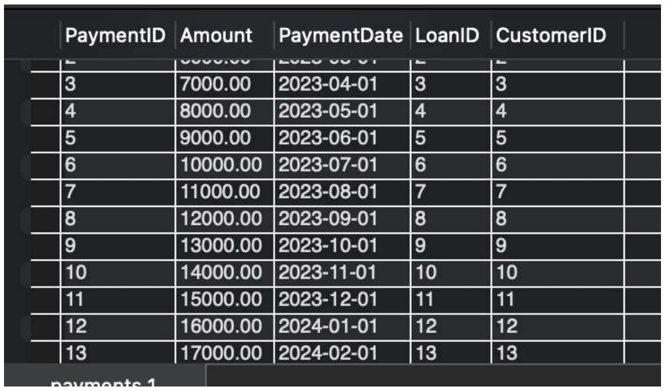
SQL : Insert statements for Payments table.

```
INSERT INTO Payments (Amount, PaymentDate, LoanID, CustomerID) VALUES
(5000.00, '2023-02-01', 1, 1),
(6000.00, '2023-03-01', 2, 2),
(7000.00, '2023-04-01', 3, 3),
(8000.00, '2023-05-01', 4, 4),
(9000.00, '2023-06-01', 5, 5),
(10000.00, '2023-07-01', 6, 6),
(11000.00, '2023-08-01', 7, 7),
(12000.00, '2023-09-01', 8, 8),
(13000.00, '2023-10-01', 9, 9),
(14000.00, '2023-11-01', 10, 10),
(15000.00, '2023-12-01', 11, 11),
(16000.00, '2024-01-01', 12, 12),
(17000.00, '2024-02-01', 13, 13),
(18000.00, '2024-03-01', 14, 14),
(19000.00, '2024-04-01', 15, 15),
(20000.00, '2024-05-01', 16, 16),
(21000.00, '2024-06-01', 17, 17),
(22000.00, '2024-07-01', 18, 18),
(23000.00, '2024-08-01', 19, 19),
(24000.00, '2024-09-01', 20, 20),
(25000.00, '2024-10-01', 21, 21),
(26000.00, '2024-11-01', 22, 22),
(27000.00, '2024-12-01', 23, 23),
(28000.00, '2025-01-01', 24, 24),
(29000.00, '2025-02-01', 25, 25);
```

Output :



```
206  16:07:36  INSERT INTO Payments (Amount, PaymentDate, LoanID, CustomerID) VALU...  25 row(s) affected Records: 25  Duplicates: 0  Warnings: 0  0.0024 sec
```

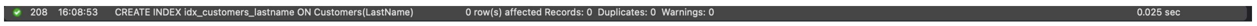


PaymentID	Amount	PaymentDate	LoanID	CustomerID
3	7000.00	2023-04-01	3	3
4	8000.00	2023-05-01	4	4
5	9000.00	2023-06-01	5	5
6	10000.00	2023-07-01	6	6
7	11000.00	2023-08-01	7	7
8	12000.00	2023-09-01	8	8
9	13000.00	2023-10-01	9	9
10	14000.00	2023-11-01	10	10
11	15000.00	2023-12-01	11	11
12	16000.00	2024-01-01	12	12
13	17000.00	2024-02-01	13	13

2. **Creating index :**
   a. **Index on LastName column of Customers table.**
      SQL :

```
CREATE INDEX idx_customers_lastname ON Customers(LastName);
```

Output :

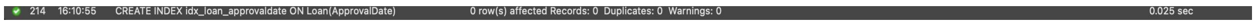


```
208   16:08:53   CREATE INDEX idx_customers_lastname ON Customers(LastName)   0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0   0.025 sec
```

   b. **Index on ApprovalDate column of Loan table.**
      SQL :

```
CREATE INDEX idx_loan_approvaldate ON Loan(ApprovalDate);
```

Output :

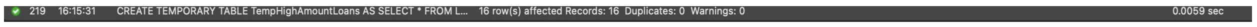


```
214   16:10:55   CREATE INDEX idx_loan_approvaldate ON Loan(ApprovalDate)   0 row(s) affected Records: 0  Duplicates: 0  Warnings: 0   0.025 sec
```

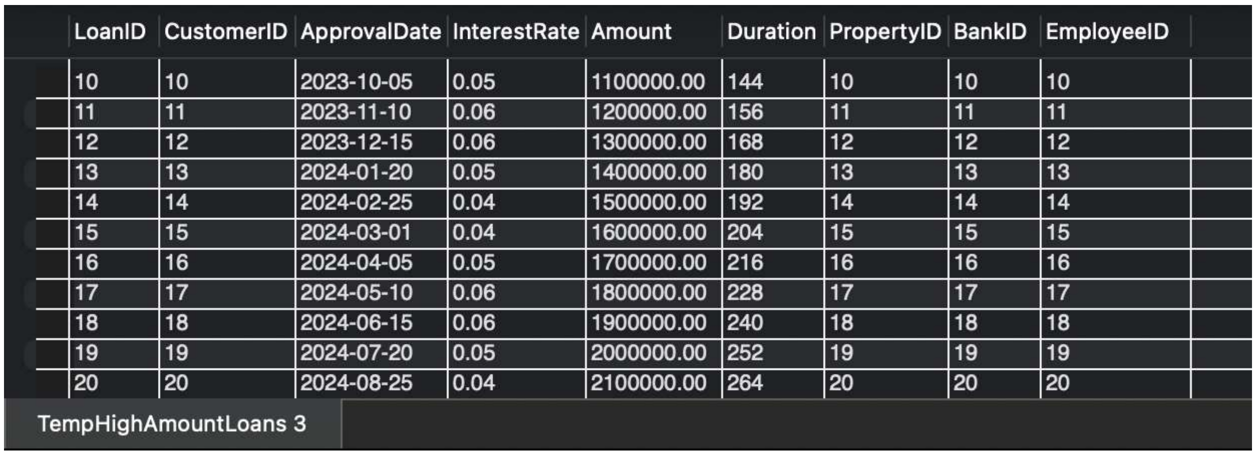
3. **Creating Temporary Table:**
   a. **Temporary table for loans with amounts greater than a specified threshold.**
      SQL :

```
CREATE TEMPORARY TABLE TempHighAmountLoans AS
SELECT *
FROM Loan
WHERE Amount >1000000;
```

Output :



```
219   16:15:31   CREATE TEMPORARY TABLE TempHighAmountLoans AS SELECT * FROM L...   16 row(s) affected Records: 16  Duplicates: 0  Warnings: 0   0.0059 sec
```

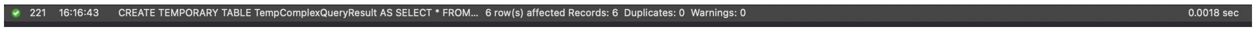


LoanID	CustomerID	ApprovalDate	InterestRate	Amount	Duration	PropertyID	BankID	EmployeeID
10	10	2023-10-05	0.05	1100000.00	144	10	10	10
11	11	2023-11-10	0.06	1200000.00	156	11	11	11
12	12	2023-12-15	0.06	1300000.00	168	12	12	12
13	13	2024-01-20	0.05	1400000.00	180	13	13	13
14	14	2024-02-25	0.04	1500000.00	192	14	14	14
15	15	2024-03-01	0.04	1600000.00	204	15	15	15
16	16	2024-04-05	0.05	1700000.00	216	16	16	16
17	17	2024-05-10	0.06	1800000.00	228	17	17	17
18	18	2024-06-15	0.06	1900000.00	240	18	18	18
19	19	2024-07-20	0.05	2000000.00	252	19	19	19
20	20	2024-08-25	0.04	2100000.00	264	20	20	20

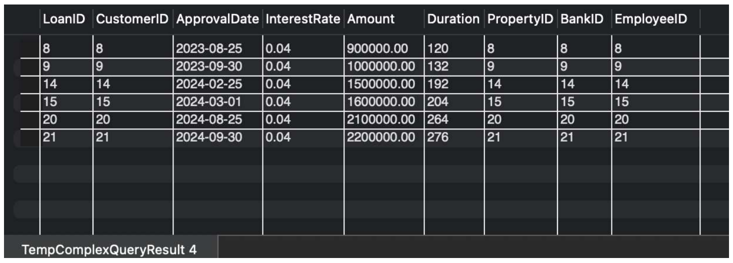
b. **Temporary table for storing the result of a complex query.**
SQL :

```
CREATE TEMPORARY TABLE TempComplexQueryResult AS
SELECT *
FROM Loan
WHERE Duration > 60 AND InterestRate<0.05;
```

Output :



```
221   16:16:43   CREATE TEMPORARY TABLE TempComplexQueryResult AS SELECT * FROM...  6 row(s) affected Records: 6  Duplicates: 0  Warnings: 0   0.0018 sec
```



LoanID	CustomerID	ApprovalDate	InterestRate	Amount	Duration	PropertyID	BankID	EmployeeID
8	8	2023-08-25	0.04	900000.00	120	8	8	8
9	9	2023-09-30	0.04	1000000.00	132	9	9	9
14	14	2024-02-25	0.04	1500000.00	192	14	14	14
15	15	2024-03-01	0.04	1600000.00	204	15	15	15
20	20	2024-08-25	0.04	2100000.00	264	20	20	20
21	21	2024-09-30	0.04	2200000.00	276	21	21	21

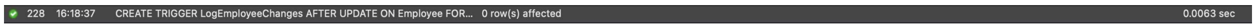
TempComplexQueryResult 4

4. **Creating Triggers :**

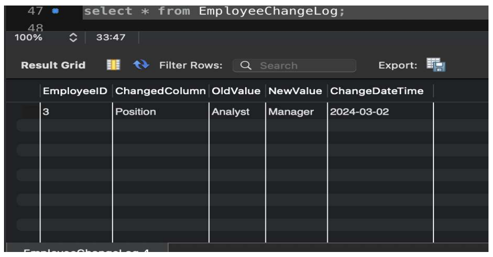
a. **Trigger to log changes made to Employee table.**
SQL :

```
create table EmployeeChangeLog (EmployeeID int , ChangedColumn varchar(20),
OldValue varchar(20), NewValue varchar(20), ChangeDateTime date);

DELIMITER //
CREATE TRIGGER LogEmployeeChanges AFTER UPDATE ON Employee
FOR EACH ROW
BEGIN
   INSERT INTO EmployeeChangeLog (EmployeeID, ChangedColumn, OldValue,
NewValue, ChangeDateTime)
   VALUES (OLD.EmployeeID, 'Position', OLD.Position, NEW.Position, NOW());
END//
DELIMITER ;
```

Output :



```
228   16:18:37   CREATE TRIGGER LogEmployeeChanges AFTER UPDATE ON Employee FOR...  0 row(s) affected   0.0063 sec
```



```
select * from EmployeeChangeLog;
```

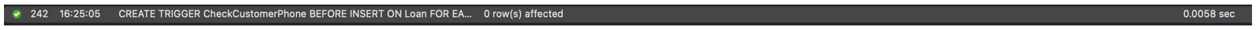
EmployeeID	ChangedColumn	OldValue	NewValue	ChangeDateTime
3	Position	Analyst	Manager	2024-03-02

b. **Trigger to enforce a constraint where a loan cannot be approved if the customers phone number is not provided.**

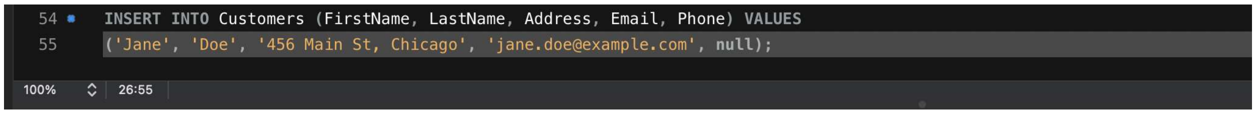
SQL :

```
DELIMITER //
CREATE TRIGGER CheckCustomerPhone BEFORE INSERT ON Loan
FOR EACH ROW
BEGIN
  DECLARE customerPhone VARCHAR(20);

  -- Retrieve the customer's phone number based on CustomerID
  SELECT Phone INTO customerPhone
  FROM Customers
  WHERE CustomerID = NEW.CustomerID;

  -- Check if the customer's phone number is null or empty
  IF customerPhone IS NULL OR customerPhone = '' THEN
    -- Raise an error if the customer's phone number is not provided
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Loan cannot be approved. Customer phone number is
required.';
  END IF;
END//

DELIMITER ;
```

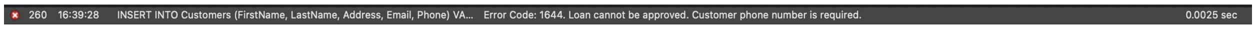
Output :



```
242  16:25:05  CREATE TRIGGER CheckCustomerPhone BEFORE INSERT ON Loan FOR EA...  0 row(s) affected  0.0058 sec
```



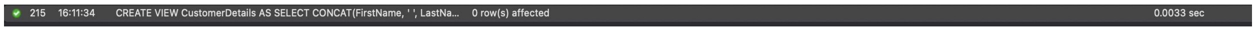
```
INSERT INTO Customers (FirstName, LastName, Address, Email, Phone) VALUES
('Jane', 'Doe', '456 Main St, Chicago', 'jane.doe@example.com', null);
```



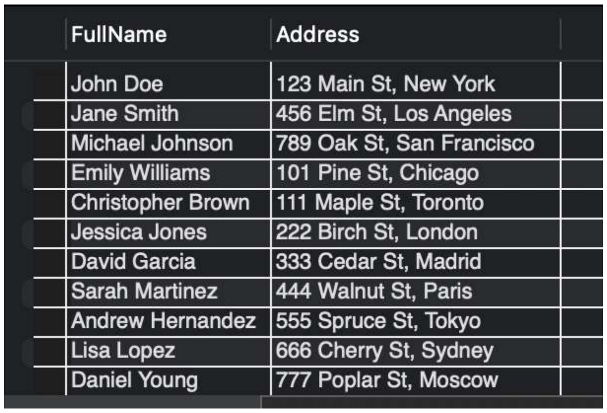
```
260  16:39:28  INSERT INTO Customers (FirstName, LastName, Address, Email, Phone) VA...  Error Code: 1644. Loan cannot be approved. Customer phone number is required.  0.0025 sec
```

5. **Creating Views:**
   a. **View to retrieve full names and addresses of customers**
      SQL :
      CREATE VIEW CustomerDetails AS
      SELECT CONCAT(FirstName, ' ', LastName) AS FullName, Address
      FROM Customers;

      Output :



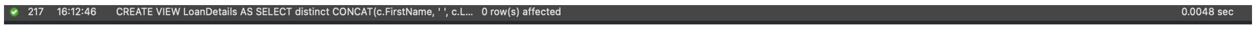
```
215   16:11:34   CREATE VIEW CustomerDetails AS SELECT CONCAT(FirstName, ' ', LastNa...   0 row(s) affected   0.0033 sec
```



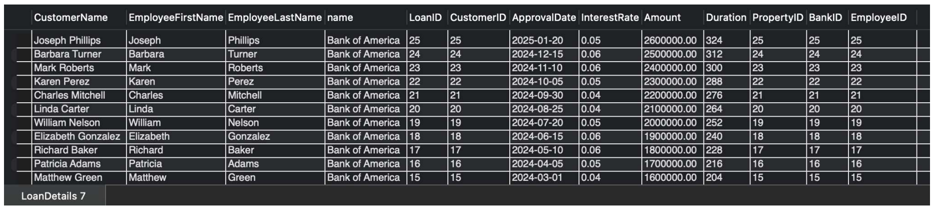
FullName	Address
John Doe	123 Main St, New York
Jane Smith	456 Elm St, Los Angeles
Michael Johnson	789 Oak St, San Francisco
Emily Williams	101 Pine St, Chicago
Christopher Brown	111 Maple St, Toronto
Jessica Jones	222 Birch St, London
David Garcia	333 Cedar St, Madrid
Sarah Martinez	444 Walnut St, Paris
Andrew Hernandez	555 Spruce St, Tokyo
Lisa Lopez	666 Cherry St, Sydney
Daniel Young	777 Poplar St, Moscow

   b. **View to retrieve loan information with associated customer and employee details.**
      SQL :
      CREATE VIEW LoanDetails AS
      SELECT distinct CONCAT(c.FirstName, ' ', c.LastName) AS CustomerName,
      e.FirstName AS EmployeeFirstName, e.LastName AS EmployeeLastName,b.name,l.*
      FROM Loan l
      JOIN Customers c ON l.CustomerID = c.CustomerID
      JOIN Employee e ON l.EmployeeID = e.EmployeeID
      JOIN Bank b on b.bankid - l.bankid;

      Output :



```
217   16:12:46   CREATE VIEW LoanDetails AS SELECT distinct CONCAT(c.FirstName, ' ', c.L...   0 row(s) affected   0.0048 sec
```



CustomerName	EmployeeFirstName	EmployeeLastName	name	LoanID	CustomerID	ApprovalDate	InterestRate	Amount	Duration	PropertyID	BankID	EmployeeID
Joseph Phillips	Joseph	Phillips	Bank of America	25	25	2025-01-20	0.05	2600000.00	324	25	25	25
Barbara Turner	Barbara	Turner	Bank of America	24	24	2024-12-15	0.06	2500000.00	312	24	24	24
Mark Roberts	Mark	Roberts	Bank of America	23	23	2024-11-10	0.06	2400000.00	300	23	23	23
Karen Perez	Karen	Perez	Bank of America	22	22	2024-10-05	0.05	2300000.00	288	22	22	22
Charles Mitchell	Charles	Mitchell	Bank of America	21	21	2024-09-30	0.04	2200000.00	276	21	21	21
Linda Carter	Linda	Carter	Bank of America	20	20	2024-08-25	0.04	2100000.00	264	20	20	20
William Nelson	William	Nelson	Bank of America	19	19	2024-07-20	0.05	2000000.00	252	19	19	19
Elizabeth Gonzalez	Elizabeth	Gonzalez	Bank of America	18	18	2024-06-15	0.06	1900000.00	240	18	18	18
Richard Baker	Richard	Baker	Bank of America	17	17	2024-05-10	0.06	1800000.00	228	17	17	17
Patricia Adams	Patricia	Adams	Bank of America	16	16	2024-04-05	0.05	1700000.00	216	16	16	16
Matthew Green	Matthew	Green	Bank of America	15	15	2024-03-01	0.04	1600000.00	204	15	15	15

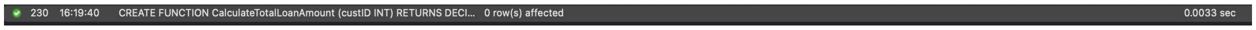
LoanDetails 7

6. **Creating Functions :**
   a. **Function to calculate total loan amount for a specific customer**

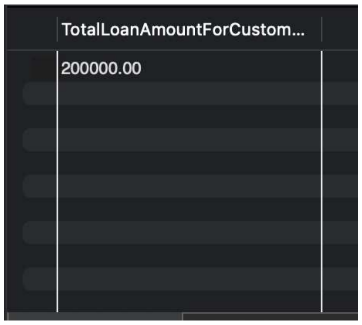
   SQL :

```
DELIMITER //
CREATE FUNCTION CalculateTotalLoanAmount (custID INT) RETURNS DECIMAL(15, 2)
READS SQL DATA
BEGIN
   DECLARE totalLoan DECIMAL(15, 2);
   SELECT SUM(Amount) INTO totalLoan
   FROM Loan
   WHERE CustomerID = custID;
   RETURN totalLoan;
END//
DELIMITER ;
```

   Output :



```
230   16:19:40   CREATE FUNCTION CalculateTotalLoanAmount (custID INT) RETURNS DECI...   0 row(s) affected   0.0033 sec
```



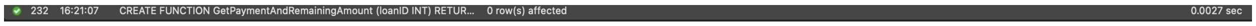
```
TotalLoanAmountForCustom...
200000.00
```

   b. **Function to determine number of payments made for a particular loan**

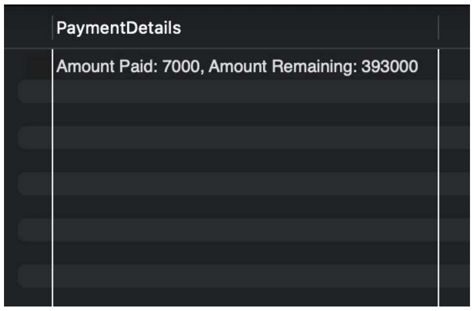
   SQL :

```
DELIMITER //
CREATE FUNCTION GetPaymentAndRemainingAmount (loanID INT) RETURNS
VARCHAR(255) READS SQL DATA
BEGIN
   DECLARE amount_paidINT;
   DECLARE amount_remainingINT;
   DECLARE result VARCHAR(255);

   SELECT SUM(p.amount), SUM(l.Amount) - SUM(p.amount) INTO amount_paid,
amount_remaining
   FROM Payments p
   INNER JOIN Loan l ON p.LoanID = l.LoanID
   WHERE p.LoanID = loanID;

   SET result = CONCAT('Amount Paid: ', amount_paid, ', Amount Remaining: ',
amount_remaining);
   RETURN result;
END//
DELIMITER ;
```

Output :



```
232   16:21:07   CREATE FUNCTION GetPaymentAndRemainingAmount (loanID INT) RETUR...   0 row(s) affected   0.0027 sec
```



```
PaymentDetails
Amount Paid: 7000, Amount Remaining: 393000
```

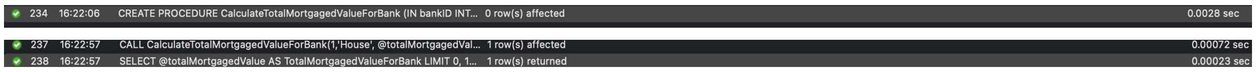
7. **Creating Stored Procedures :**
   a. **Procedure that calculates the total value of properties mortgaged by a specific bank:**

SQL :

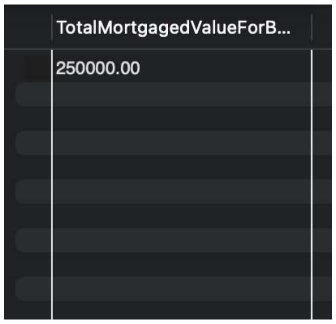
```
DELIMITER //
CREATE PROCEDURE CalculateTotalMortgagedValueForBank (IN bankID INT, IN
Type_Value VARCHAR(20) , OUT totalMortgagedValue DECIMAL(15,2))
BEGIN
   -- Declare variable to store total mortgaged value
   DECLARE total DECIMAL(15,2);

   -- Calculate total mortgaged value for the bank
   SELECT COALESCE(SUM(Value), 0) INTO total
   FROM  Loan l, Property p
   WHERE l.BankID = bankID
   AND l.Propertyid = p.propertyid
   AND Type = Type_Value;

   -- Assign the result to the OUT parameter
   SET totalMortgagedValue = total;
END//
```

DELIMITER ;

Output:



```
234   16:22:06   CREATE PROCEDURE CalculateTotalMortgagedValueForBank (IN bankID INT...  0 row(s) affected      0.0028 sec
237   16:22:57   CALL CalculateTotalMortgagedValueForBank(1,'House', @totalMortgagedVal...  1 row(s) affected   0.00072 sec
238   16:22:57   SELECT @totalMortgagedValue AS TotalMortgagedValueForBank LIMIT 0, 1...  1 row(s) returned   0.00023 sec
```



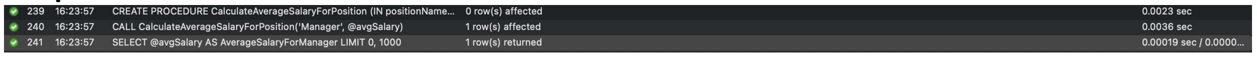
```
TotalMortgagedValueForB...
250000.00
```

b. **Procedure that retrieves the average salary of employees within a specific position:**

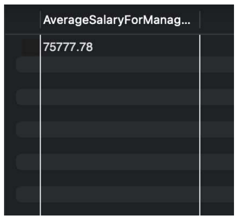
SQL :

```
DELIMITER //
CREATE PROCEDURE CalculateAverageSalaryForPosition (IN positionName
VARCHAR(50), OUT averageSalary DECIMAL(10,2))
BEGIN
   -- Declare variable to store average salary
   DECLARE avgSalary DECIMAL(10,2);

   -- Calculate average salary for employees in the specified position
   SELECT COALESCE(AVG(Salary), 0) INTO avgSalary
   FROM Employee
   WHERE Position = positionName;

   -- Assign the result to the OUT parameter
   SET averageSalary = avgSalary;
END//
DELIMITER ;
```

Output :



```
239   16:23:57   CREATE PROCEDURE CalculateAverageSalaryForPosition (IN positionName...   0 row(s) affected   0.0023 sec
240   16:23:57   CALL CalculateAverageSalaryForPosition('Manager', @avgSalary)            1 row(s) affected   0.0036 sec
241   16:23:57   SELECT @avgSalary AS AverageSalaryForManager LIMIT 0, 1000               1 row(s) returned   0.00019 sec / 0.0000...
```



```
AverageSalaryForManag...
75777.78
```